

ARRAYÜZLER(INTERFACE)

Öğr.Gör. Murat ASLANYÜREK

ARRAYÜZ KAVRAMINA GİRİŞ

- Arayüzler, soyut sınıflara benzer. Ancak yapı olarak bazı yönleriyle farklılıklar gösterir.
 - Arayüzlerde bütün metodlar gövdesiz olarak tanımlanırlar.
 - Arayüzler, sınıfların bir işi nasıl yapacağını değil, işi yaparken hangi adımları veya ne yapması gerektiğini tanımlar.
 - Arayüzler çoklu kalıtım olayını basite indirmek için oluşturulmuştur.
 - Arayüzlerin sınıfları birleştirme özelliği vardır.
 - Sınıflar kullanmak istediği arayüzü sınıf tanımından sonra **implements** anahtar sözcüğü ile kendi bünyelerine dahil ederler.
 - Arayüzler içerisinde tanımlanmış erişim belirleyicilerin **public static** ve **final** tipindedir ve ilk değer ataması zorunludur.(Arayüzü kullanacak diğer sınıflar, değişkenleri değiştiremezler)

ARRAYÜZ KAVRAMINA GİRİŞ

- Sınıflar birden fazla arayüzü bünyesine katabilir.
- **Not:** Arayüzler, soyut sınıflarda olduğu gibi hem gövdeli hem gövdesiz metodlara sahip değildir. Tamamen gövdesiz metodlar bulunur. Yani bu metodlar başlı başına bir iş yapamazlar. Bu metodların implement edilen sınıflarda override edilmeleri gerekir.

calisan.java

```
package Arayuzler;

public interface calisan { // arayüz tanımlaması
    double oran = 0.7;
    double ucret () ;
    void calisanBolumu () ;
    void ucretBelirle ( double ucretSabit) ;
    // arayüzü kullanacak olan sınıflarda tanımlanacak metodlar
}
```

- **Not:** Arayüzün metodlarını, arayüzü kullanacağımız sınıflarda override ederek yeniden tanımlamak zorundayız.

isci.java

```
1 package Arayuzler;
2
3 public class isci implements calisan{
4     double ucretSabiti;
5
6     //Çalışan ücretini döndüren metod
7     @Override
8     public double ucret(){
9         return ucretSabiti*oran;
10    }
11
12    //çalışan bölümünü ekrana yazdıran metod
13    @Override
14    public void calisanBolumu(){
15        System.out.println("Ucretli Calisan");
16    }
17
18    // çalışan ücretini belirtmek için kullanılacak metod
19    @Override
20    public void ucretBelirle(double ucretSabiti){
21        this.ucretSabiti=ucretSabiti;
22    }
23 }
```

mudur.java

```
1 package Arayuzler;
2
3 public class mudur implements calisan {
4     double ucretSabiti;
5
6     @Override
7     public double ucret(){
8         return ucretSabiti;
9     }
10
11    @Override
12    public void calisanBolumu(){
13        System.out.println("Yönetici");
14    }
15
16    @Override
17    public void ucretBelirle(double ucretSabiti){
18        this.ucretSabiti= ucretSabiti;
19    }
20 }
```

satisElemani.java

```
1 package Arayuzler;
2 public class satisElemani implements calisan{
3     double ucretSabiti;
4     double komisyon;
5     final double komisyonOrani=0.3;
6
7     @Override
8     public double ucret(){
9         return ((ucretSabiti*oran)+komisyon);
10    }
11    @Override
12    public void calisanBolumu(){
13        System.out.println("Satis Elemanı");
14    }
15    @Override
16    public void ucretBelirle(double UcretSabiti){
17        this.ucretSabiti=ucretSabiti;
18    }
19    public void satisKomisyonuBelirle(int adet){
20        komisyon = adet*komisyonOrani;
21    }
22 }
```

arayuzOrnegi.java

```
1 package Arayuzler;
2
3 public class arayuzOrnegi {
4     public static void main(String[] args){
5         isci ucretliCalisan = new isci();
6         ucretliCalisan.calisanBolumu();
7         ucretliCalisan.ucretBelirle(865.70);
8         System.out.println("Maaşı : "+ ucretliCalisan.ucret());
9
10        mudur yoneticisi = new mudur();
11        yoneticisi.calisanBolumu();
12        yoneticisi.ucretBelirle(5519.71);
13        System.out.println("Maaşı : "+ yoneticisi.ucret());
14
15        satisElemani pazarlama = new satisElemani();
16        pazarlama.calisanBolumu();
17        pazarlama.ucretBelirle(5519.71);
18        System.out.println("Maaşı : "+ pazarlama.ucret());
19    }
20
21 }
```

Console Problems Debug Shell Search

<terminated> arayuzOrnegi [Java Application] C:\Program Files\Java\jdk-13.0.1\bin\javaw.exe (13

Ucretli Calisan
Maaşı : 605.99
Yönetici
Maaşı : 5519.71
Satıs Elemanı
Maaşı : 0.0

- **Not:** Arayüzler, soyut sınıflara benzediğinde aralarında önemli farklar vardır. Eğer her metodun override edilmesini istiyorsak, bu metodları bir arayüz içerisinde tanımlarız. Ayrıca arayüzlerde, soyut sınıflarda olduğu gibi bir ilişki kavramı yoktur. Yani arayüz ve bunu kullanan sınıflar arasında kalıtım açısından bir bağlantı olmayabilir.

ARRAYÜZLERDE GENİŞLETME İŞLEMİ

- Tanımlanan bir arayüzü genişletebiliriz. Bunu bir arayüzün başka bir arayüzü kalıtım yoluyla devralmasıyla mümkün olur.
- **Not:** Bir arayüz içerisinde aynı isimde iki metod varsa , bu metodların aldığı parametrelerin farklı olması gerekir. Dönüş tiplerinin farklı olması bir anlam ifade etmez, hata oluşur. Bu metodlardaki overload işlemine benzer.

Canli.java

```
1 package Arayuzler;
2
3 public interface Canli {
4     public void nefesAl();
5 }
6 interface Hayvan extends Canli{
7     public void avlan();
8 }
9 interface Surungen extends Hayvan{
10     public void surun();
11 }
12 class Tavsan implements Hayvan{
13     @Override
14     public void nefesAl(){
15         System.out.println("Nefes Aliyor");
16     }
17     public void avlan(){
18         System.out.println("Avlanıyor");
19     }
20     class Timsah implements Surungen{
21         public void avlan(){
22             System.out.println("Avlanıyor");
23         }
24         public void surun(){
25             System.out.println("Surunuyor");
26         }
27         public void nefesAl(){
28             System.out.println("Nefes Aliyor");
29         }
30     }
31 }
32
```

Bu örneğimizde 3 adet arayüz tanımladık. Bu arayüzlerden Hayvan arayüzünden Tavsan sınıfını implement ettik. Surungen arayüzünden de Timsah sınıfını implement ettik. Tavsan sınıfımız, Hayvan ve Canli arayüzlerindeki metodları override etmek zorunda kaldı. Çünkü Hayvan arayüzü de Canli arayüzünden türetilmiştir (miras).

Timsah sınıfımız ise, hem Hayvan hem Canli hemde Surungen arayüzlerindeki metodları override etmek zorunda kaldı. Bunun sebebi de Sürungen arayüzünün hayvan arayüzünden, Hayvan arayüzünün ise Canli arayüzünden miras alınmasıdır.

ARRAYUZ İÇERİSİNDE BAŞKA BİR ARRAYÜZ KULLANMA

- Bir arrayüz başka bir arrayüz içerisinde bulunabilir.

```

1 package Arayuzler;
2
3 interface Arayuz1{
4
5     interface Arayuz2{
6         public void metod2();
7     }
8
9     public void metod1();
10 }
11
12 interface Arayuz3{
13     public void metod3();
14 }
15
16 class Sinif1 implements Arayuz1{
17     public void metod1() {
18         System.out.println("metod1 override edildi");
19     }
20 }
21
22
23 class Sinif2 implements Arayuz1.Arayuz2{
24     public void metod2() {
25         System.out.println("metod2 override edildi");
26     }
27 }
28
29 class Sinif3 implements Arayuz3{
30     public void metod3() {
31         System.out.println("metod3 override edildi");
32     }
33 }

```

```

34 public class iciceArayuzCalistir {
35     public static void main(String args[]) {
36         Sinif1 s1 = new Sinif1();
37         s1.metod1();
38         Sinif2 s2 = new Sinif2();
39         s1.metod1();
40         Sinif3 s3 = new Sinif3();
41         s3.metod3();
42     }
43 }

```

 Console
  Problems
  Debug Shell
  Search

```

<terminated> iciceArayuzCalistir [Java Application] C:\Program Files\Java\jdk-13.0
metod1 override edildi
metod1 override edildi
metod3 override edildi

```

Örnek Uygulama(a)

- Aşağıdaki arayüz iki tane metot içermektedir.

```
1 package Arayuzler;  
2  
3 public interface arayuz01 {  
4  
5     public void topla(int x, int y);  
6  
7     public void hacim(int x, int y, int z);  
8 }
```

Örnek Uygulama(b)

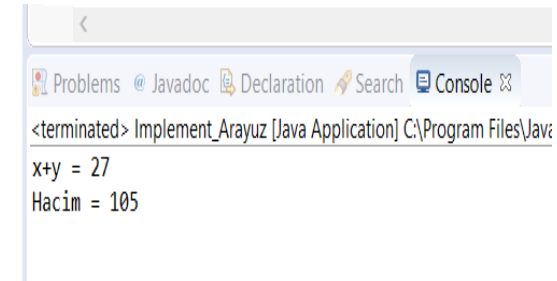
- Aşağıdaki arayüz iki tane sabit değişken içermektedir.

```
1 package Arayuzler;  
2  
3 public interface arayuz02 {  
4     public static final double fiyat = 1250.00;  
5     public static final int sayac = 5;  
6 }
```

Örnek Uygulama(c)+(a+b)

- Aşağıdaki sınıf, önceki iki arayüzü var etmektedir. Sınıf içinde, arayüzün metotlarının serbestçe tanımlandığına dikkat ediniz. Aynı arayüzü var edecek başka sınıflar, bu metotları başka başka tanımlayabilirler. Bu, bir metodun farklı işler görmesini sağlar ve polymorphism diye bilinir.

```
1 package Arayuzler;  
2 public class Implement_Arayuz implements arayuz01, arayuz02 {  
3  
4     public void toplama(int x, int y) {  
5         System.out.println("x+y = " + (x + y));  
6     }  
7  
8     public void hacim(int a, int b, int c) {  
9         System.out.println("Hacim = " + (a * b * c));  
10    }  
11  
12    public static void main(String[] args) {  
13        Implement_Arayuz d01 = new Implement_Arayuz();  
14        d01.toplama(12, 15);  
15        d01.hacim(3, 5, 7);  
16    }  
17 }
```



The screenshot shows an IDE console window with the following content:

```
<terminated> Implement_Arayuz [Java Application] C:\Program Files\Java  
x+y = 27  
Hacim = 105
```

Ödev

1. İçerisinde alan ve çevre hesaplayan metodlar bulunan bir arayüz tanımlayın. Bu arayüzü oluşturacağınız üçgen, dikdörtgen, kare ve daire sınıfları kullansın. Bu sınıflar alan ve çevrelerini bu metodlar yardımıyla hesaplayıp ekrana yazdırsın.
2. İçerisinde türü, yazarı, fiyatı ve yazdır () isimli bir metodu bulunan bir arayüz yazın. Bu arayüzü Bilimkurgu, Polisiye ve Macera sınıfları kullansın. Her biri kendi bilgilerini ekrana yazdırabileceği bir sınıfı yazın.